(2)

Fractal Image Encoding

SBIR Phase II
Quarterly Progress Report
Contract #N00014-91-C-0117

Yuval Fisher
Albert Lawrence
NETROLOGIC, Inc.

March 18, 1992

To: Office of Naval Research • Dr M. Schlesinger • 800 N. Quincy St. • Arlington, VA 22217-5000

From: NETROLOGIC, Inc. • 5080 Shoerham Place • Suite 201 • San Diego, CA 92122

Summary

From November 1991 to January 1992 the focus of the work was the development of IFS algorithms and computer code for image compression, the refinement of algorithms and computer code for compressing color images and the development of wavelet methods for image compression.

Our improvements in compression algorithm have shortened image compression times significantly. During this time period two papers were submitted for publication.

During February and March we held discussions with Larry Domash, who heads a group at Foster-Miller. Dr. Domash is the PI of a Phase II SBIR on optical computing methods funded at Foster-Miller by SDIO and managed by ONR (Wm. Michelli). One subject of the Foster-Miller program is the development of optical processing methods for IFS encoding. Because the subjects of our two ONR SBIR programs are related, Dr. Domash discussed the availability of a possible collaboration with Mr. Michelli and subsequently approached Netrologic. The objective of the subsequent discussions has been to determine whether optical methods, particularly those associated with four-wave mixing, could be used to shorten image compression times.

Image Decompression

Rapid image decompression is crucial for many potential Navy and commercial users of the fractal image compression scheme. During the previous report period we developed a completely new algorithm for decoding images.

When decoding during the iterative method, an initial image $f_{\{0\}}$ in L^2 on $[0,1] \times [0,1]$ is used to compute the iterates $f_{\{n\}} = W\left(f_{\{n-1\}}\right)$, where

$$W(\bullet) = \bigcup_i W_i(\bullet). \tag{1}$$

This can also be written as

$$f_{\{n\}}(x,y) = \sum_i a_i f_{\{n-1\}}\left(V^{-1}_i(x,y)\right) + o_i, \tag{2}$$

where the ith term of the summation is defined on the region R_i, V_i is an affine transformation on the region R_i, and a_i and o_i are contrast and intensity adjustments respectively.

Suppose we are decompressing an image of $M \times M$ pixels. We can write the image as a column vector, and then the above equation can be written

$$f_{\{n\}} = S f_{\{n-1\}} + O \tag{3}$$

where S is an $M^2 \times M^2$ matrix with entries s_{ij} which encode the contrast adjustments a_i and spatial affine transformations V_i and O is a column vector encoding the intensity adjustments. Then

$$f_{\{n\}} = S^n f_{\{0\}} + \sum_i^{n-1} S^i O. \tag{4}$$

If each s_{ij} is less than 1 then the first term is 0 in the limit. This condition can be relaxed if **W** is eventually contractive. When **I - S** is invertible

$$f_{\infty} = \sum_{i}^{\infty} S^{i} O = (I - S)^{-1} O. \qquad (5)$$

If each pixel value of $f_{\{n\}}$ depends upon only one (or even a few) pixel values of $f_{\{n-1\}}$ then $I - S$ is very sparse and is inverted readily by sparse matrix methods. We have implemented this algorithm and it shows promising results. It runs considerably faster than the previous version of the iteration algorithm.

The new decoding triangularizes the S matrix by pivoting. We have observed that different pivoting schemes lead to different decoding times since each pivot operation eliminates an entry in one place in the matrix and creates another somewhere else. If the new entry is located above the diagonal, no further pivot is required, but if it is located below, another pivot will be required. We are studying dereferencing schemes which allow several simultaneous pivot operations in the case where one pivot requires another.

In a separate effort, we are also implementing integer versions of the iteration scheme with a variant on the storage of the transforms. In the iterative scheme, each pixel depends upon another (with a brightness and a contrast adjustment). Rather than computing this dependency from the transformations at each iteration step, we compute it initially. This also appears to lead to considerable improvement and to times which rival the matrix inversion method. Current times are 8 or less on 486 machines about 20 seconds on 286 machines without a co-processor.

Encoding Optimization

The encoding algorithm consists of searching through a large number of sub-images, in order to find one which has a low RMS error when correlated with a piece of a quadtree partition of the image. In order to reduce this search time, we classify the collections of sub-images. The search is limited to elements of the same class, leading to a reduced search time.

Our classification is as follows:

> Each sub-image is first partitioned into 4 equal quadrants, which are ordered by brightness. This ordering leads to 3 classes, once a rotation and a flip operation are used to bring the brightest quadrant into the upper left position. This gives three classes and also determines a rotation and a flip operation (reducing the search class by a factor of 8, the number of symmetry operations on the square). These three classes are further split into 24 by ordering the contrast levels of the sub-quadrants of each quadrant. This gives a total of 72 classes.

We are evaluating this improvement. Initial results show a 50-fold increase in encoding speeds.

Color Image Encoding

We have investigated alternative methods of color encoding. Typically, color images are stored as indices to a color look-up table. This means that an 8 bit per pixel image can contain 256 colors from a palette which contains (typically) 2^{24} colors. Encoding such images is done usually by expanding the data to a red-green-blue (RGB) representation, converting this representation to a YIQ representation (also used in color television transmission) and encoding the three signals separately. The advantage of this scheme is that the I and Q signals can be stored very compactly with very little perceptual degradation. The disadvantage is that three images must be encoded.

This algorithm has been implemented. The implementation involves some tricky color look-up table manipulation because screen displays have a palette of 256 colors, while the YIQ signals typically yield a much larger number of colors. To circumvent this problem, we store the color map of the original image along with the compressed image. The display process finds the color from the table that is closest to the YIQ color from the decompression.

We also have investigated an alternative scheme which depends on the internal ordering of the color look-up table. The color entries are ordered first by hue and within each hue class by the Y value. The difficulty is finding the hue classes. The Y-ordering within each hue class means that the color image represented by the color lookup table entries will share some geometric features with the Y channel only. Difficulties arise at hue class boundaries, as well as in defining the hue classes. Although initial experiments appear very promising, the artifacts arising from boundary cross-over have been difficult to eliminate. Even when the hue class index is stored with the image, so that the hue class of each pixel is known, the difficulty in automatically segregating the colors results in poor image quality.

We are continuing to investigate this method. While difficult and complicated, the potential utility of such an algorithm is wide and is not restricted to fractal methods. In particular because only one channel is encoded (vs three) there is a very large time and space savings.

Optical methods

Optical four wave mixing can be used to construct the convolution correlation of four images. This operation may be represented by

$$I_r = I_1 \bullet (I_2 * I_3), \tag{6}$$

where I_r, I_1, I_2, and I_3 denote the output and the three input images respectively, while $\bullet$ represents the correlation operation and $*$ represents the convolution operation.

The major bottleneck in computing an IFS code for an image is checking the large set of possible linear transforms and windows. In particular, it is necessary to compute

$$I_w = T(W \cdot I) \bullet I, \tag{7}$$

for a large number of linear transforms, T, and windows, W. In this formula, $\cdot$ represents multiplication while $\bullet$ represents correlation In particular $W \cdot I$ is the windowed image corresponding to a tile R_i in some partition as described above.

In order to put this into the form of a three wave product we use the convolution theorem:

$$W \cdot I = F^{-1}(FW^{*}FI), \tag{8}$$

where F and F^{-1} correspond to the Fourier transform and inverse Fourier transform respectively. Applying Formula (8) to Formula (7) we obtain

$$I_w = TF^{-1}(FW^{*}FI) \bullet I. \tag{9}$$

The composition of operators TF^{-1} can also be written $F^{-1} T_F$, where T_F is the linear transform induced by T in the spatial frequency domain.

We are currently discussing the implementation of the operation described by Equation (9) by means of optical techniques with Larry Domash of Foster Miller.